Cuadernos de Documentación Multimedia
ISNN: 1575-9733

Modificación de campos y atributos de un gestor de base de datos por medio de un documento de texto

Jorge Caldera Serrano

Recibido: 14.5.2010

Aprobado: 12.10.2010

Resumen

Se muestra un método para la fácil modificación de una base de datos creada con el lenguaje de programación C# y el lenguaje de compilación Visual C #, sin que para ello deba tenerse conocimientos de programación ni de implementación de bases de datos, siento tan fácil como la modificación de un documento de texto con una serie de parámetros sencillos como son la indicación del tipo de campo, número de líneas que ocupa, nombre interno y nombre en la base de datos. Para ello no sólo se muestra el método de modificación sino que se aporta el Código Fuente del programa sobre el que se desarrolla los cambios. El realizar la programación en el lenguaje C# de la Plataforma .NET abre la posibilidad de utilizarlo tanto para software comercializados como bajo plataforma libre como Linux. Este desarrollo, creado inicialmente para una base de datos de imagen fija para televisión, cuenta con gran flexibilidad y adaptabilidad de modificación que lo valida para cualquier ámbito informativo-documental.

Palabras claves

Base de Datos, C #, Documento de Imagen Fija , Usuario inexperto, Implementación y Diseño

Abstract

In this article it is showed the methodology to make an easy alteration of a Database Management System created under the use of the C# programming language and the compilation language Visual C#, without having any database programming knowledge or implementing database knowledge, being easier as the alteration of a text document with a series of simple parameters as the kind of field instruction, number of lines took up, internal name and the database name. For that purpose, it is showed, not only the alteration method but the program source code about the changes developed. The reason for using the C# language programming (.NET Plattform) open the possibility to use it both for proprietary software systems and free software, as Linux. This development, initially created for a television still-picture database, counts on big alteration flexibility and adaptability that validates it for any informative-research field.

Keywords

Database, C #, Still-Picture Document, Inexperienced User, Implementation and Design

CONTEXTUALIZACIÓN Y OBJETIVO DE LA INVESTIGACIÓN

La investigación en Information Science viene normalmente marcada por un estudio práctico de la realidad realizado generalmente por medio de la observación. En nuestra área la práctica profesional ha sido la que ha hecho evolucionar nuestra disciplina, al tener que buscar soluciones a problemas reales de algunos sistemas de información de gestión documental, y por medio de la solución de problemas inicialmente parciales se ha podido extrapolar los resultados a otros sistemas de información.

La Investigación Básica ha sucumbido a la Investigación Práctica entre nuestros investigadores y/o docentes y por supuesto por los desarrollos de entidades de carácter privado. Dicho factor que inicialmente debiera ser un problema para la evolución y desarrollo de cualquier disciplina, en Information Science derivado probablemente de su juventud, hace posible que muchas de las aplicaciones encuentren validez no sólo para la aplicación que se desarrolló sino también para otras materias.

Estimamos que nuestra trabajo puede contar con dicha valía, la de ser válida para múltiples ámbitos, y dentro del Information Science para diferentes entornos empresariales.

Nuestras investigaciones se centran en la gestión, tratamiento, conservación y difusión de la documentación en los medios de comunicación, centrándonos en los medios audiovisuales, y en el estudio matriz de este resultado, se encuentra el estudio de la gestión de la documentación fotográfica en las cadenas televisivas. Entre otros datos, se observa que muchas cadenas (aquellas de nueva creación o con escasez de personal) derivan a profesionales inexpertos o sin estudios adecuados la gestión de la documentación fotográfica centrándose sus profesionales en la información audiovisual.

Por lo tanto, e identificada dicha realidad, se cree oportuno la creación por medio de la programación de un gestor de base de datos que sea inicialmente válido y fácil de implementar y modificar por estos agentes inexpertos, sin conocimientos sobre organización de información ni de programación en bases de datos, ni de los conocimientos más simples, como pudiera ser la lógica booleana.

Por lo tanto, en esta parte del desarrollo se desea alcanzar el siguiente objetivo: que cualquier usuario y/o gestor de información pueda crear la estructura de una base de datos sin conocimientos informáticos, sirviéndose para dar las órdenes al programa de un documento en formato de texto. En definitiva, se desea que el usuario pueda “programar” y organizar la información gestionada desde una Base de Datos sin conocimientos previos y tan sólo utilizando como intefaz un fichero de texto para proyectar su herramienta.

Volvemos a recordar que este estudio está encuadrado en los agentes de los medios de comunicación para el tratamiento de información fotográfica, pero se estima que es igualmente válido para otros gestores y unidades informativas.

Deseamos aportar no sólo sus ventajas sino también mostrar el código fuente de dicha herramienta para que pueda ser volcado. Por motivos de extensión de programación obviamos otra serie de elementos como todos los interfaz gráficos de la herramienta como elementos propios creados para la documentación fotográfico, centrándonos en la capacidad de

programar y cambiar la estructura de la base de datos con gran simplicidad, y todo ello pudiendo ser realizado con software comercializado y software libre.

METODOLOGÍA

No deseamos extendernos en la metodología por la que se ha llevado a cabo dicha labor derivada de su simpleza: dentro de un estudio mayor observamos que muchos gestores de información carecen de la formación más elemental sobre base de datos, y a partir de ahí se programa una herramienta ágil, fácil, ergonómica y amigable, y siempre pensando en la simplicidad.

Por lo tanto, y volviendo a recordar que este resultado es una parte de otro proyecto de mayor envergadura, señalar que la labor ha sido desarrollada partiendo de una creencia a todas luces insalvable en la actualidad: la multidisciplinariedad de nuestra disciplina y la transversalidad de conocimientos necesarios para desarrollar componentes técnicos.

Por ello, esta labor cuenta con un elevando componente de las Ciencias de la Computación ya que ha sido necesario el programar una visión documental en una rutina informática. Muchas veces nuestros deseos son de difícil adaptación por parte de la computación, y en muchos casos los errores provienen de un difícil entendimiento entre dos tipos de profesionales complementarios pero a la vez alejados (los gestores de información y los programadores de software); sin embargo se ha conseguido llegar a conseguir resultados positivos, según nuestra valoración.

Tras los pertinentes contactos, se comienza a trabajar con el lenguaje de Programación C # (C SPAN) y como lenguaje compilador Visual C #, los cuales serán analizados con posterioridad brevemente, para desarrollar las premisas expuestas y contando con la hipótesis de que si existen herramientas de más fácil, mejor y mayor implementación, será más opimo el tratamiento de información.

En un primer momento, el resultado que mostramos es un fichero de texto y no un gestor de base de datos, la cual debe ser implementada utilizando, por ejemplo, MySQL, ampliando así su rendimiento en cuanto al número de ítems analizables. Dicha fichero de texto podría llegar a contar con unas 10.000 fotografías gestionadas, con la inclusión de MySQL las posibilidades se multiplican exponencialmente.

El método de exposición de resultado es la identificación del código fuente de una rutina de programación que, tal y como se ha contextualizado, se ha creado para una base de datos de naturaleza fotográfica (imagen fija). Además se explicación cómo deben realizarse las modificaciones.

Requerimientos y Herramientas

Las herramientas básicas de programación han sido el motor de la plataforma de Microsoft .Net, el lenguaje de programación C# y el lenguaje de compilación Visual C#. Para estar familiarizado con la programación orientada a objetos es interesante tener conocimientos previos de programación C++ o Java, de los cuales deriva C#.

Microsoft.NET es un conjunto de nuevas tecnologías con el objetivo de obtener una plataforma sencilla para distribuir software que puedan ser suministrados remotamente y que puedan comunicarse y combinarse de manera independiente a la plataforma.

El Common Language Runtime (CLR) es el motor encargado de gestionar las aplicaciones, simplificando los desarrollos y favoreciendo la fiabilidad y seguridad. Algunas de las características son el ser un modelo de programación consistente sobre el que se podrá acceder a diferentes servicios; ser un modelo de programación sencillo, al desaparecer elementos complejos incluidos en los sistemas operativos actuales; ser ejecutable en multiplataforma al actuar como máquina virtual, por lo que puede ejecutarse bajo Microsoft y Linux; es válido para integrar lenguajes creados bajo la plataforma .NET; detecta errores de programación difíciles de localizar; cuenta con una distribución transparente, aportando la infraestructura necesaria para crear objetos remotos y acceder a ellos de manera transparente y cuenta con seguridad avanzada al proporcionar mecanismos para restringir la ejecución de ciertos códigos según su procedimiento de usuario que se ejecute.

C# es el lenguaje de propósito general destinado para la plataforma .NET. Sus creadores son Scott Wiltamuth y Anders Hejlsberg; aunque son muchos los lenguajes que pueden utilizarse, éste es específico para la plataforma.

La sintaxis y estructura de C# es muy similar a C++, ya que se desea facilitar la migración de los códigos y el aprendizaje de programación. Toma las mejores características de lenguajes anteriores (Visual Basic, Java o C++) combinándolas.

Las principales características de C# son su sencillez en la programación, su modernidad al incorporar elementos que se han demostrado útil para el desarrollo de aplicaciones, orientada a objetos no admitiendo ni funciones ni variables globales sino que todo el código y datos han de definirse dentro de tipos de datos, orientada a componentes ya que incluye elementos orientados al diseño de componentes que otros lenguajes tiene que simular (permite definir propiedades cómodamente), gestión automática de memoria, seguridad de tipos incluyendo mecanismos que permite asegurar los accesos a tipos de datos, versionable ya que permite crear nuevas versiones de tipos sin temor a que la introducción de nuevos miembros provoquen errores difíciles de detectar, siendo además eficiente y compatible.

Resultado de la programación

La programación realizada ha sido la que se expone a continuación:

```
using System;
using System.Collections;
using System.Text;
using System.IO;

namespace GestioFotos.Datos
{
	/// <summary>
	/// Descripción breve de BDFotografia.
	/// </summary>
	public class BDFotografia
	{
		private ArrayList _busqueda=new ArrayList();
		private ArrayList _totalidad=new ArrayList();
		public BDFotografia()
		{
			_busqueda.Clear();
			_totalidad.Clear();
			try
			{
				using (StreamReader sr=new StreamReader(Aplicacion.datos.fichero,Encoding.Default))
				{
					string linea=sr.ReadLine();
					while(linea=="[#FOTOGRAFIA#]")
					{
						Fotografia foto=new Fotografia();
						linea=leerFotografia(sr,foto);
						_totalidad.Add(foto);
					}
				}
			}
			catch(Exception e)
			{
				Console.WriteLine("No se puede leer el fichero de BD:");
				Console.WriteLine(e.Message);
			}
		}
		~BDFotografia()
		{
			try
			{
				using (StreamWriter sw=new StreamWriter(Aplicacion.datos.fichero,false,Encoding.Default))
				{
					foreach(Fotografia foto in _totalidad)
						escribirFotografia(sw,foto);
					sw.WriteLine("[#FINAL#]");
				}
			}
			catch (Exception e)
			{
				Console.WriteLine("No se puede escribir el fichero de BD:");
				Console.WriteLine(e.Message);
			}
		}
		void escribirFotografia(StreamWriter sw,Fotografia foto)
		{
			sw.WriteLine("[#FOTOGRAFIA#]");
			foreach(Fotografia.Dato informacion in foto.valores)
			{
				sw.WriteLine("["+informacion.nombre+"]"+informacion.informacion);
			}
```

```
}
string leerFotografia(StreamReader sr,Fotografia foto)
{
	string linea=sr.ReadLine();
	while(linea!="[#FINAL#]" && linea!="[#FOTOGRAFIA#]")
	{
		string clave,valor;
		clave=linea.Substring(1,linea.IndexOf("]")-1);
		valor=linea.Substring(linea.IndexOf("]")+1);
		foto[clave]=valor;
		linea=sr.ReadLine();
	}
	return linea;
}
public void busquedaCrear(Fotografia mas,Fotografia menos)
{
	_busqueda.Clear();
	foreach(Fotografia foto in _totalidad)
	{
		bool anyadir=true;
		foreach(string clave in Fotografia.posibles)
	  {
	    if(mas[clave]!=null)
	    {
	      if(foto[clave]==null)
	      {
	        anyadir=false;
	      }
	      else
	      {
	                                    if(foto[clave].IndexOf(mas[clave])==-1)
	                                    {
	                                        anyadir=false;
	                                    }
	      }
	    }
	    if(menos[clave]!=null)
	    {
	                                if(foto[clave]!=null)
	                                {
	                                    if(foto[clave].IndexOf(menos[clave])!=-1)
	                                    {
	                                        anyadir=false;
	                                    }
	                                }
	    }
	  }
		if(anyadir)
			_busqueda.Add(_totalidad.IndexOf(foto));
	}
}
public int busquedaIndice(int indiceBusqueda)
{
	if(_busqueda.Count<=indiceBusqueda)
		return -1;
	return (int)_busqueda[indiceBusqueda];
}
public Fotografia busquedaElemento(int indiceBusqueda)
{
	int indiceTotalidad=busquedaIndice(indiceBusqueda);
	if(indiceTotalidad==-1)
		return null;
	return (Fotografia)_totalidad[indiceTotalidad];
}
public int totalBusqueda
{
	get
```

```
                {
                        return _busqueda.Count;
                }
        }
        public void borraImagen(int indice)
        {
                Fotografia foto=busquedaElemento(indice);
                _totalidad.Remove(foto);
        }
        public void analizaFicheros(string[] ficheros)
        {
                foreach(string fichero in ficheros)
                        analizaFichero(fichero);
        }
        private void analizaFichero(string fichero)
        {
                string carpeta="."+fichero.Remove(0,Aplicacion.datos.directorio.Length);
                foreach(Fotografia foto in _totalidad)
                {
                        if(foto.fichero==carpeta)
                                return;
                }
                Fotografia fotoNueva=new Fotografia();
                fotoNueva["CARPETA"]=carpeta;
                FileInfo info=new FileInfo(fichero);
                fotoNueva["MIDA"]=((info.Length+1023)/1024).ToString();
                _totalidad.Add(fotoNueva);
        }
        private void analizaDirectorio(string directorio)
        {
                string[] ficheros=Directory.GetFiles(directorio);
                foreach(string fichero in ficheros)
                {
                        if(Path.GetExtension(fichero).ToLower()==".jpg" ||
                                        Path.GetExtension(fichero).ToLower()==".tif")
                        {
                                analizaFichero(fichero);
                        }
                }
                string[] directorios=Directory.GetDirectories(directorio);
                foreach(string direct in directorios)
                {
                        analizaDirectorio(direct);
                }
        }
        public void analizaDirectorio()
        {
                analizaDirectorio(Aplicacion.datos.directorio);

        }
    }
}
```

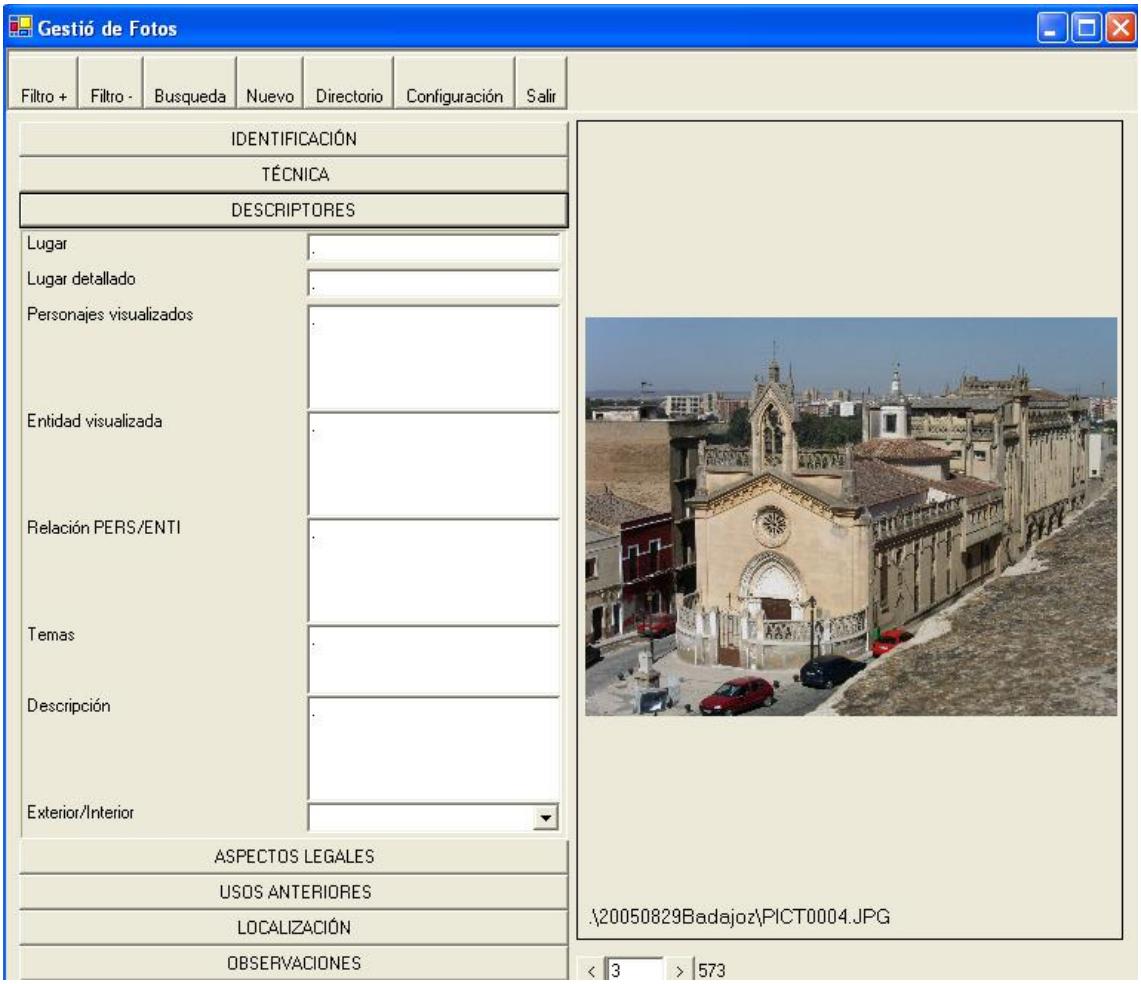
MÉTODO DE MODIFICACIÓN

Lo importante de esta programación es su amigabilidad a la hora de modificar o llevar a cabo una nueva estructuración de la base de datos.

Inicialmente la base puede contar con una estructura que se refleja en un documento que puede abrirse con cualquier procesador de texto, como Word pad o el procesador de Windows Word. La estructura que aparece a continuación, y sacado de un documento en Word, no va a ser analizada ya que simplemente nos sirve de ejemplo para explicar la fórmula de modificación así como su contenido y forma de elaboración que posteriormente se indicará.

```
l,IDENTIFICACIÓN
        d,1,FECHA ENTRADA,Fecha entrada
        ls
        ca,1,PROGRAMA,Programa
        ca,1,SUBPROGRAMA,Subprograma
        ls
        d,1,FECHA ACTO,Fecha noticia
        a,2,TITULO,Título
        a,1,LOCALIDAD,Localidad
l,TÉCNICA
        n,1,MEDIDAS,Medidasida
        c,1,FORMAT,Formato,JPEG;TIF
        c,1,V_H,V/H,Vertical;Horizontal
        c,1,PLANO,Plano,General;Americano;medio;primero;primerísimo;detalle
        c,1,CALIDAD,Calidad,nítida;Borrosa;Color pujat
        c,1,SOPORTE,Soporte,Informàtic;Paper;Diapositiva
        a,1,ARXIU,Otros soportes
l,DESCRIPTORES
        a,1,LUGAR,Lugar
        a,1,ESPACIO,Lugar detallado
        a,3,PERSONA,Personajes visualizados
        a,3,ENTIDAD,Entidad visualizada
        a,3,RELACION,Relación PERS/ENTI
        a,2,TEMAS,Temas
        a,3,DESCRIPCION,Descripción
        c,1,EXTERIOR,Exterior/Interior,Exterior;Interior
```

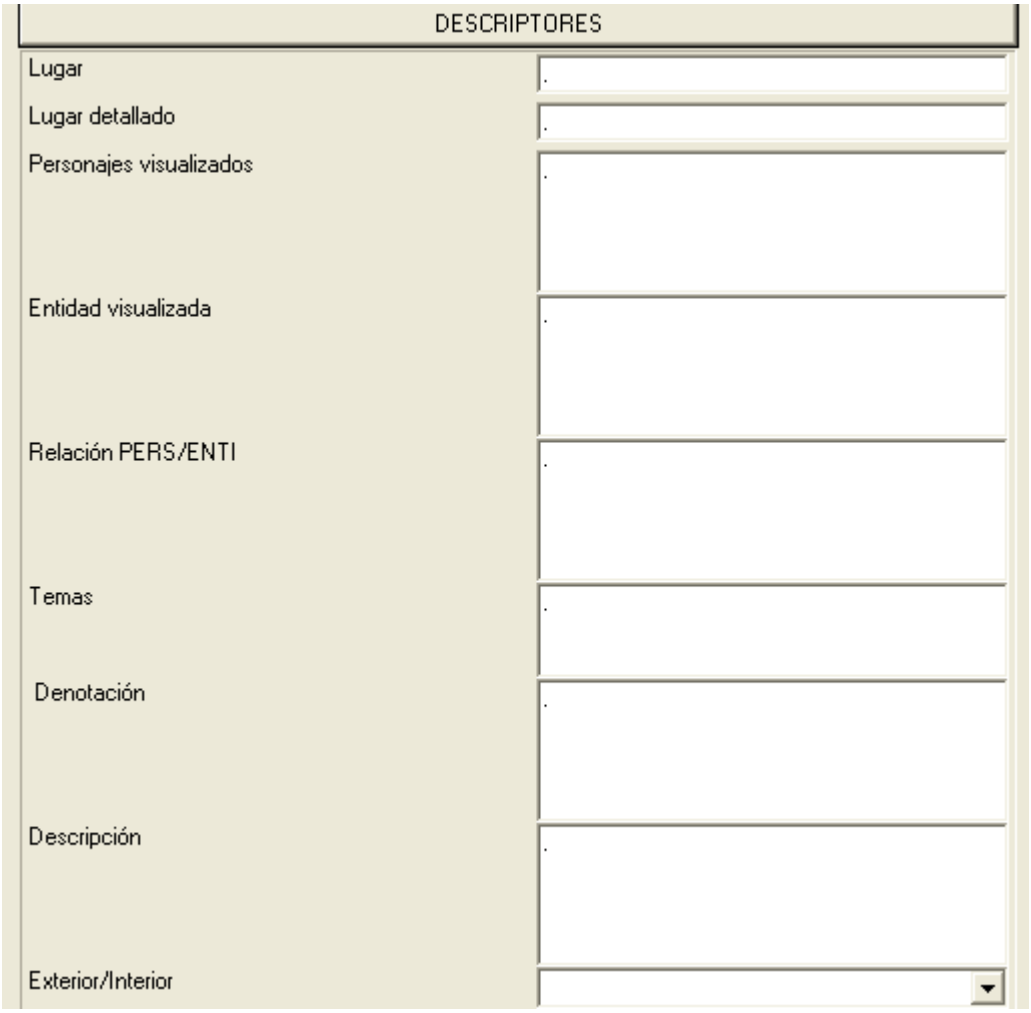
En este ejemplo se han incluido tres áreas de la base de datos, cada línea corresponde a un campo de la base de datos, que quedaría visualmente tal y como sigue:



La forma de inclusión sería tan sencillo como incluir en una nueva línea (tal y donde se quiera que aparezca) la orden:

```
I,DESCRIPTORES
        a,1,LUGAR,Lugar
        a,1,ESPACIO,Lugar detallado
        a,3,PERSONA,Personajes visualizados
        a,3,ENTIDAD,Entidad visualizada
        a,3,RELACION,Relación PERS/ENTI
        a,2,TEMAS,Temas
        a,3,DENOTACIÓN TEMÁTICA,  Denotación
        a,3,DESCRIPCION,Descripción
        c,1,EXTERIOR,Exterior/Interior,Exterior;Interior
```

Dando como resultado la siguiente tabla, tras guardar la nueva modificación por medio de la orden “Salvar Archivo”, sin modificar su nombre y ubicación.



Como puede observarse su simplicidad necesita de unos mínimos conocimientos (no de base de datos) sino de ordenación de la información que está repartida en cuatro bloques informativos:

- Tipo de datos
- Línea/s que ocupa
- Nombre interno
- Nombre externo

a,3,DENOTACIÓN TEMÁTICA, Denotación

En la información que hemos incluido anteriormente se indica.

- Tipo de datos: (a): alfabético.
- Línea/s que ocupa: 3
- Nombre interno: DENOTACIÓN TEMÁTICA
- Nombre externo: Denotación.

Los posibles tipos de datos que pueden utilizarse son:

- (a): alfabético
- (n): numérico
- (d): fecha
- (c): desplegable con valores
- (ca): desplegables con valores fijados
- (x): caja de verificación o checkbox

POSIBILIDADES Y LÍNEAS FUTURAS

Como puede apreciarse las posibilidades que ofrece este método de implementación, e incluso de diseño, de una base de datos cuenta con una hipótesis inicial que se convierte en su punto fuerte: la facilidad de utilización.

No siempre los profesionales de la información cuentan con un bagaje informacional y formativo válido para poder desarrollar una base de datos por lo que esta herramienta se convierte en un útil y válido utensilio de gestión de bases de datos. De todas maneras, la simplificación la hace útil aunque se tengan conocimientos de gestión y/o programación de bases de datos.

Sin lugar a dudas otra de sus grandes ventajas es ser una base de datos muy portable, que podría ser ejecutada bajo cualquier sistema operativo, abriendo así la posibilidad a utilizarlo bajo software libre lo que estimamos una línea de futuro tanto en investigación como en forma de actuación en el ámbito de las Ciencias de la Información, motivado por su bajo coste y su rápido desarrollo por un colectivo de investigadores más unido y con una comunicación fluida.

La capacidad de modificación tan fácilmente lo hace especial, atractivo y útil, mientras que lo innecesario de conocimientos previos lo vuelve eficaz. Además, con las mutaciones de nuestros usuarios y de las estructuras de las bases de datos, parece oportuno crear herramientas que puedan ser modificadas con facilidad.

Estimamos que son muchas las posibilidades con las que cuenta este desarrollo para la gestión de bases de datos programadas con C# y portable a cualquier sistema operativo, desde el momento que deben desarrollarse herramientas ergonómicas y útiles para usuarios no siempre especializado, sobre cuando la tendencia en muchas servicios de información es que el gestor de información no sea un guardián de información sino un gestor de conocimiento el cual pone a disposición información, herramientas y útiles para que pueda ser autosuficiente ante la gran cantidad de información.

BIBLIOGRAFÍA

Ashenfelter, JP. (1999). Choosing a database for your web site. Nueva York [etc.]: John Wiley and Sons.

Auffret, G. & Bachimont, B. (1999) Audiovisual cultural heritage: from TV and radio archiving to hypermedia publishing. Research and advanced technology for digital libraries, proceedings lecture notes in computer science, Vol. 1698, 58-75.

Cochrane, C (1999). Video database systems: issues, products and applications. Journal of Documentation, 55, 4, 460-461.

Chorafas, D.N. (1994). Intelligent multimedia databases: from object orientation and fuzzy engineering to intentational data base structure. New Jersey: Englewood Cliffs.

Connolly, T.M.; et. al. (1995). Database systems : a practical approach to design, implementation and management. Wokingham: Addison-Wesley.

De Vries, Ap (2001). Content independence in multimedia databases. Journal of the American Society for Information Science and Technology 52 (11):954-960.

De Vries Ap, Windhouwer M, Apers PMG, Kersten M (2000). Information access in multimedia databases based on feature model. New Generation Computing 18, 4, 323-339.

Delsemme, P. (1993) Audiovisual documentation – the problems of staging for television. Degres-Revue de Synthese a orientation semiologique, Vol. 73, B3-B11.

Edmondson, Ray (2004). Audiovisual Archiving: philosophy and principles (CI/2004/WS/2). Paris: Unesco.

Enser, P (2004). Content-based video retrieval: A database perspectiva. Journal of Documentation 60 (5): 586-588

Fidel, R. (1986). Towards expert systems forthe selection of search key. Journal of the American Society for Information Science. 37, 1, 37-44.

Gunnerson, Eric (2000). A programmer´s introduction to C#. Berkeley: Apressz

Harrison, H. (ed.) (1997). Audiovisual archives: a practical reader (CII.97/WS/4), UNESCO, Paris.

Ingwersern, P. (1992). Cognitive perspectives of information retrieval interaction: elements of a cognitive IR theory. Journal of documentation, 52, 1. 3-50.

Joint, N. (2001). Designing interfaces for distributed electronic collections: the lesons of traditional librarianship. Libri, 51, 148-156.

Kowalski, G. (1997) Information retrieval systems: theory and implementation. Boston: Kluwer.

Liberty, Jesse (2001). Programming C#. Sebastopol: O´Reilly, 2001.

Saade, G. (2004). Lebanese television archives. Electronic Library, 22, 2, 139-143.

Tam, Am & Leung CHC (2001). Structured natural-lenguage descriptions for semantic content retrieval of visual material. Journal of the American Society for Information Science and Technology, 52, 11, 930-937.

Troelsen, Andrew (2001). C# and the .NET Framework. Berkeley: Apress, 2001